Self-Supervised Monocular Scene Flow Estimation
– Supplementary Material –

Junhwa Hur Stefan Roth
Department of Computer Science, TU Darmstadt

In this supplementary material, we provide further details on the learning rate schedules, data augmentation, and the hyper-parameter settings. Afterwards, we provide a more comprehensive study of the decoder design, qualitative examples for the loss ablation study, and a qualitative comparison with the state-of-the-art Mono-SF approach [3].

A. Learning Rate Schedule

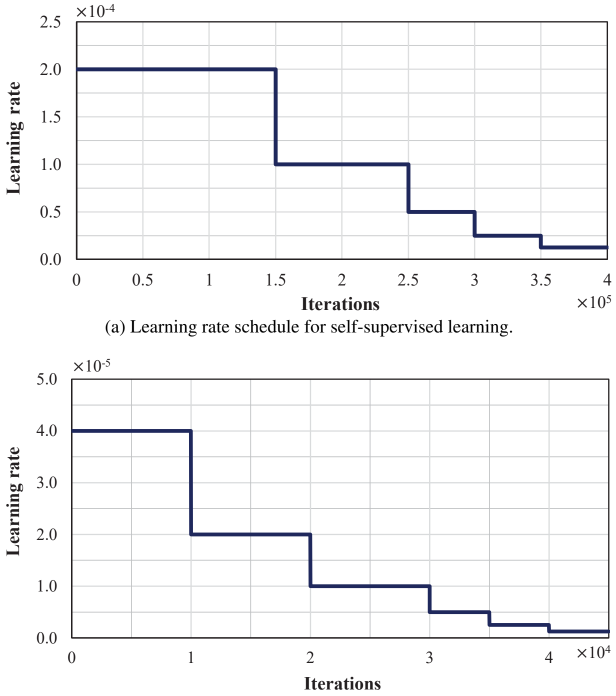
Fig. 6 illustrates the learning rate schedules for both self-supervised learning and semi-supervised fine-tuning. When first training our model in a self-supervised manner for 400k iterations, the initial learning rate starts from 2×10^{-4} and is halved at 150k, 250k, 300k, and 350k iteration steps. When fine-tuning in a semi-supervised manner afterwards, the training schedule consists of 45k iterations; the initial learning rate starts from 4×10^{-5} and is halved at 10k, 20k, 30k, 35k, and 40k iteration steps.

B. Details on Data Augmentation

As discussed in the main paper, we perform photometric and geometric augmentations at training time. Here we provide more details on our augmentation setup for both self-supervised training and semi-supervised fine-tuning.

Augmentations for self-supervised training. We apply photometric augmentations with 50% probability. Specifically, we adopt random gamma adjustments, uniformly sampled from $[0.8, 1.2]$, brightness changes with a multiplication factor that is uniformly sampled in $[0.5, 2.0]$, and random color changes with a multiplication factor that is uniformly sampled in $[0.8, 1.2]$ for each color channel.

For geometric augmentations, we first randomly crop the input images with a random scale factor uniformly sampled in $[93\%, 100\%]$ and apply random translations uniformly sampled from $[-3.5\%, 3.5\%]$ w.r.t. the input image size. Then we resize the cropped image to 256 × 832 pixels as in previous work [27, 28, 30, 40, 60]. We also apply a horizontal flip [26, 27, 28, 53] with 50% probability. Because the geometric augmentations have an effect on the camera intrinsics, we adjust the intrinsic camera matrix accordingly by calculating the corresponding camera center and focal length of each augmented image. At testing time, we only resize the input image to 256 × 832 pixels without photometric augmentation.



(a) Learning rate schedule for self-supervised learning.

(b) Learning rate schedule for fine-tuning.

Figure 6. **Learning rate schedules** for *(a)* self-supervised learning and *(b)* semi-supervised fine-tuning.

Augmentations for semi-supervised fine-tuning. Likewise, we also apply the same photometric augmentations with 50% probability. For geometric augmentations, we only apply random cropping without scaling and then resize to 256 × 832 pixels. Not performing scaling is to avoid changes to the ground truth, which may happen if zooming and interpolating the sparse ground truth. The crop size $s \cdot h_0 \times s \cdot w_0$ is determined by the cropping factor s that is uniformly sampled in $[94\%, 100\%]$, where h_0 and w_0 is height and width of the original input resolution. At testing time, the same augmentation scheme as during self-supervised training applies: resizing the input images to 256×832 pixels without photometric augmentation. However, we note that better augmentation protocols can likely be discovered with further investigation [65].

C. Hyper-Parameter Settings

Our self-supervised proxy loss in Eq. (1) of the main paper has a total of 6 hyper-parameters, which could make it difficult to achieve satisfactory results without careful tuning. In this section, we thus discuss how we choose the hyper-parameters and provide an analysis on how sensitive the scene flow accuracy is depending on the hyper-parameter choices.

First, as discussed in the main paper, the balancing weight λ_{sf} between the two joint tasks in Eq. (1) is dynamically determined to make the loss of the scene flow and disparity be equal in every iteration [21]. For the disparity loss, we simply adopt the same hyper-parameters (*i.e.*, $\lambda_{\text{d\_sm}}$, α, and β in Eqs. (2), (3b) and (4), respectively) as in previous work [13], which leaves only two hyper-parameters, $\lambda_{\text{sf\_sm}}$ and $\lambda_{\text{sf\_pt}}$, to tune in the scene flow loss, Eq. (5). We perform grid search on the two parameters.

Table 8 gives the grid search results regarding the two hyper-parameters, reporting the accuracy for monocular depth, optical flow, and scene flow. In the upper half of the table, we fix the smoothness parameter $\lambda_{\text{sf\_sm}}$ and control the 3D point reconstruction loss parameter $\lambda_{\text{sf\_pt}}$ to see its effect on the accuracy. The bottom half of the table is set up the other way around. Note that the lower the better for all metrics.

We find that $\lambda_{\text{sf\_pt}}$ is important for best scene flow accuracy, specifically settings that yield accurate disparity information on the target frame, *D2-all*. This observation follows our design of the 3D point reconstruction loss, which penalizes the 3D distance between corresponding points, encouraging more accurate 3D scene flow in 3D space. However, as a trade-off, having a higher value of $\lambda_{\text{sf\_pt}}$ leads to lower accuracy for 2D estimation, *i.e.* of depth and optical flow. On the other hand, we find that the parameter for the 3D smoothness loss, $\lambda_{\text{sf\_sm}}$, does not strongly affect the accuracy in general. That is, once $\lambda_{\text{sf\_pt}}$ is in the right range, the results are not particularly sensitive to the parameter choice.

		Depth	Flow	Scene Flow			
$\lambda_{\text{sf\_sm}}$	$\lambda_{\text{sf\_pt}}$	Abs Rel	EPE	D1-all	D2-all	F1-all	SF1-all
1	0.005	**0.104**	7.118	**30.50**	51.48	**22.32**	62.97
	0.05	0.107	**7.057**	32.56	49.45	22.33	61.27
	0.1	0.109	7.319	33.65	**35.57**	22.58	**47.46**
	0.5	0.117	8.259	33.91	36.24	25.18	48.72
10	0.005	**0.105**	**6.934**	**31.18**	52.29	**22.15**	63.47
	0.2	0.108	7.421	31.37	**34.39**	22.73	**46.08**
	0.3	0.110	7.379	31.91	34.42	23.79	47.10
	0.4	0.113	7.773	32.79	35.53	23.98	47.63
200	0.005	**0.103**	**6.883**	**30.48**	50.05	**22.65**	61.47
	0.1	0.108	7.525	31.49	46.50	23.38	59.17
	0.2	0.107	7.197	31.40	**34.75**	23.02	**46.95**
	0.4	0.114	7.435	33.35	35.56	24.30	48.25
0.1	0.005	0.106	6.839	31.47	52.20	22.39	63.70
1		**0.104**	7.118	**30.50**	51.48	22.32	62.97
10		0.105	6.934	31.18	52.29	**22.15**	63.47
100		0.105	**6.723**	31.15	**51.05**	22.18	**62.55**
1	0.2	0.109	**7.118**	31.81	34.95	23.01	46.82
10		0.108	7.421	31.37	**34.39**	**22.73**	**46.08**
100		0.108	7.386	**31.05**	34.95	22.88	47.08
200		**0.107**	7.197	31.40	34.75	23.02	46.95
10	0.4	0.113	7.773	32.79	35.53	23.98	47.63
100		**0.111**	**7.365**	32.97	**34.63**	**23.92**	**47.29**
200		0.114	7.435	33.35	35.56	24.30	48.25
300		0.112	7.833	**31.97**	35.20	25.39	48.48

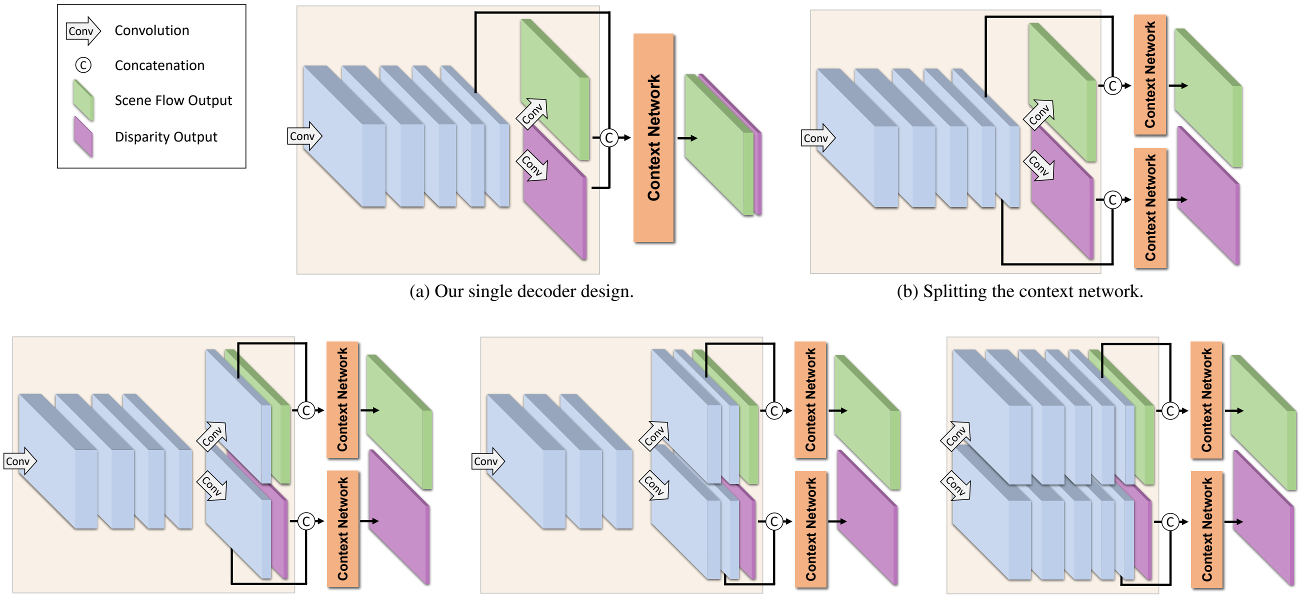
Table 8. **Grid search results on the two hyper-parameters, $\lambda_{\text{sf\_sm}}$ and $\lambda_{\text{sf\_pt}}$** based on the accuracy of monocular depth, optical flow, and scene flow. The 3D point reconstruction parameter $\lambda_{\text{sf\_pt}}$ contributes to more accurate disparity information on the target frame, *D2-all*, yielding more accurate scene flow *SF1-all* in the end. The overall results are not very sensitive to the choice of the 3D smoothness parameter $\lambda_{\text{sf\_sm}}$.

D. In-Depth Analysis of the Decoder Design

With the decoder ablation study in Table 3 of the main paper, we demonstrate that having separate decoders for disparity and scene flow yields instable, unbalanced outputs in contrast to having our proposed single decoder design. For a more comprehensive analysis, we conduct an empirical study by gradually splitting the decoder consisting of 5 convolution layers and studying the behavior of the networks for each configuration. Our backbone network, PWC-Net [45], has context networks at the end of the decoder, which are fed the output and the last feature map from the decoder as input and perform post-processing for better accuracy. In our splitting study, we also separate the context networks for each separated decoder so that the two decoders at the end of the networks do not share information.

Fig. 7 illustrates each configuration. From our single decoder design in Fig. 7a, we first split the context network for disparity and scene flow respectively, as shown in Fig. 7b. Then, we begin to split the decoder from the last convolution layer (*i.e.*, Fig. 7c), the 2$^{\text{nd}}$-to-last layer (*i.e.*, Fig. 7d), and so on until eventually completely splitting into two separate decoders (*i.e.*, Fig. 7e). To ensure the same network capacity, we adjust the number of filters so that all configurations have network parameter numbers in a similar range. All configurations are trained on the *KITTI Split* of KITTI raw [10] in our self-supervised manner.

Table 9 shows the disparity, optical flow, and scene flow accuracy of each configuration on *KITTI Scene Flow Train-*



(a) Our single decoder design.

(b) Splitting the context network.

(c) Splitting from the last layer.

(d) Splitting from the 2^{nd}-to-last layer.

(e) Splitting into two separate decoders.

Figure 7. **Gradually splitting the single decoder into two separate decoders**: we gradually split the single decoder *(a)* by first splitting the context network *(b)*, and then splitting from the last layer of the decoder *(c)*, the 2^{nd}-to-last layer *(d)*, and so on until completely splitting into two separate decoders*(e)*. For ease of visualization, we omit showing the *convolution* operation between the neighboring feature maps in the decoder.

ing [36, 37]. We first observe that splitting the context network yields a significant 32.73% decrease in scene flow accuracy (*i.e.*, SF1-all), which mainly stems from the less accurate disparity estimates (*i.e.*, D1-all and D2-all) although the optical flow accuracy remains almost the same. This provides an important outlook: given the same optical flow accuracy, the scene flow accuracy depends crucially on how well one can decompose the optical flow cost volume into depth and scene flow, where using the single decoder model works better. When further splitting the decoder starting from the last convolution layer, the networks *(i)* cannot be trained stably anymore, *(ii)* output trivial solutions for the disparity, and *(iii)* even decrease the optical flow accuracy. This observation again confirms the benefits of using our proposed single decoder design in terms of both accuracy and training stability.

E. Qualitative Analysis of Loss Ablation Study

Configuration	D1-all	D2-all	F1-all	SF1-all
Single decoder	**31.25**	**34.86**	**23.49**	**47.05**
Splitting the context network	44.19	45.02	23.51	62.45
Splitting at the last layer	100	97.22	26.46	100
Splitting at the 2^{nd}-to-last layer	100	97.22	26.39	100
Splitting at the 3^{rd}-to-last layer	100	97.22	26.94	100
Splitting at the 4^{th}-to-last layer	100	97.22	28.68	100
Splitting into two separate decoders	100	97.22	27.63	100

Table 9. **Scene flow accuracy of each decoder configuration**: splitting the context network already decreases the scene flow accuracy by 32.73%. Further splitting the decoder yields training instability with trivial solutions for the disparity output.

Table 2 in the main paper provides an ablation study of our self-supervised proxy loss. For better understanding of how each loss term affects the results, we provide qualitative examples of disparity, optical flow, and scene flow estimation. Fig. 8 displays the results for each loss configuration: *(a)* the basic loss where only the brightness and smoothness terms are active; *(b)* with occlusion handling, which discards occluded pixels in the loss; *(c)* with the 3D point reconstruction loss; and *(d)* the full loss. Each configuration is trained in the proposed self-supervised manner using the *KITTI Split* and evaluated on *KITTI Scene Flow Training* [36, 37].

Without the 3D point reconstruction loss for scene flow (*i.e.*, columns *(a)* and *(b)* in Fig. 8), the networks output inaccurate disparity information for the target frame *(D2)* especially in the road area, which yields inaccurate scene flow results *(SF1)* in the end. Applying the 3D point reconstruction loss but without occlusion handling (*i.e.*, column *(c)* in Fig. 8) results in inaccurate estimates and some artifacts appearing on out-of-bound pixels, still leading to an unsatisfactory final scene flow accuracy. These artifacts happen when the 3D point reconstruction loss tries to minimize the 3D Euclidean distance between incorrect pixel correspondences, such as for occlusions or out-of-bound pixels. Discarding those occluded regions in the proxy loss eventually yields better estimates in the occluded region as well.

F. Qualitative Comparison

We provide some qualitative examples of our monocular scene flow estimation by comparing with the state-of-the-art Mono-SF method [3], which uses an integrated pipeline of CNNs and an energy-based model. Figs. 9 and 10 show successful qualitative results as well as some failure cases of our fine-tuned model on the KITTI 2015 Scene Flow public benchmark [36, 37], respectively.

In Fig. 9, our model outputs more accurate disparity and optical flow estimation results than Mono-SF [3] without using an explicit planar surface representation or a rigid motion assumption, which would be beneficial for achieving better accuracy on the KITTI 2015 Scene Flow public benchmark.

Fig. 10, in contrast, shows some of the failure cases, where our model outputs less accurate results for scene flow estimation than Mono-SF [3]. Although our model can estimate optical flow with an accuracy comparable to Mono-SF, inaccurate disparity estimation eventually leads to less accurate scene flow. The gap in terms of the disparity accuracy of ours *vs*. Mono-SF [3] can be explained by the fact that Mono-SF exploits over 20 000 instances of pseudo ground-truth depth data to train their monocular depth model, while our method uses only 200 images for fine-tuning.

References

[65] Aviram Bar-Haim and Lior Wolf. ScopeFlow: Dynamic scene scoping for optical flow. In *CVPR*, 2020. 2

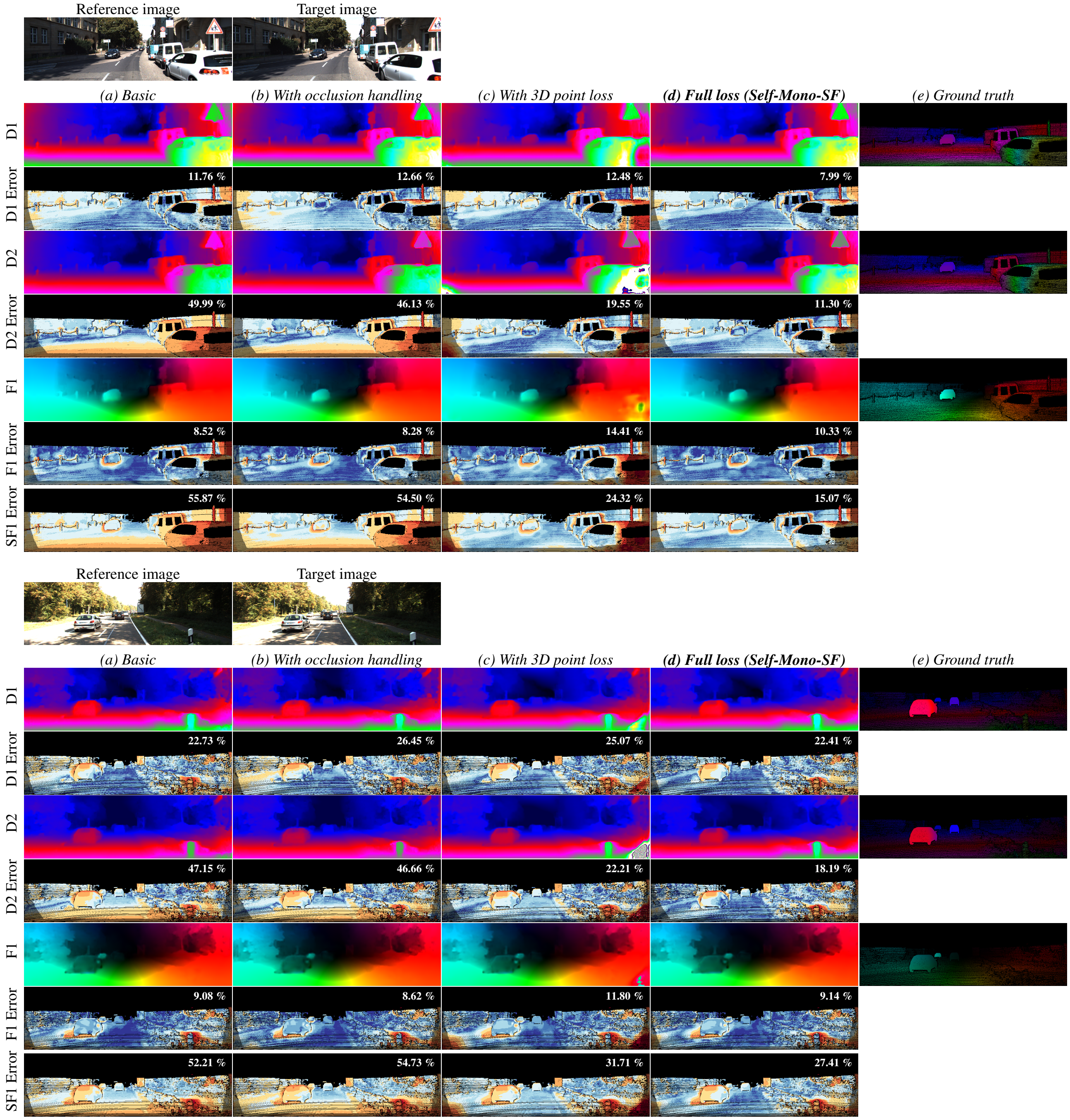


Figure 8. **Qualitative examples on the loss ablation study**. For each scene in the first row we show two input images, the reference and the target image. From the second to the last row, we show a qualitative comparison of each loss configuration: *(a)* basic loss, *(b)* with occlusion handling, *(c)* with 3D point reconstruction loss, and the *(d)* our full loss. Each row visualizes the disparity map of the reference image *(D1)* with its error map *(D1 Error)*, disparity estimation at the target image mapped into the reference frame *(D2)* along with its error map *(D2 Error)*, optical flow *(F1)* with its error map *(F1 Error)*, and the scene flow error map *(SF1 Error)*. The outlier rates are overlayed on each error map. The last column shows *(e)* the ground truth for each estimate.

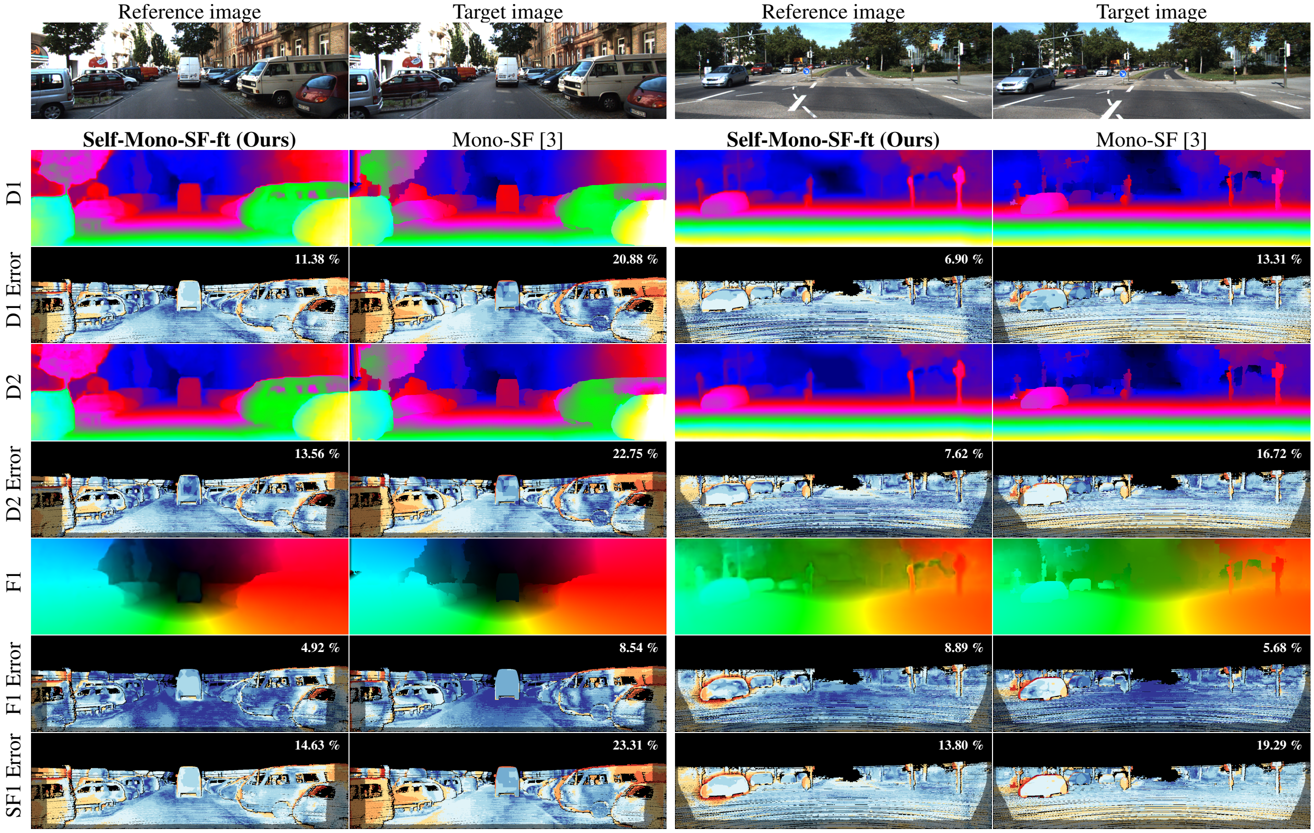


Figure 9. **Some successful cases and qualitative comparison with the state of the art on the KITTI 2015 Scene Flow public benchmark [36, 37]**. In the first row, we show two input images, the reference and target image. From the second to the last row, we give a qualitative comparison with Mono-SF [3]: the disparity map of the reference image *(D1)* with its error map *(D1 Error)*, disparity estimation at the target image mapped into the reference frame *(D2)* along with its error map *(D2 Error)*, optical flow *(F1)* with its error map *(F1 Error)*, and the scene flow error map *(SF1 Error)*. The outlier rates are overlayed on each error map.

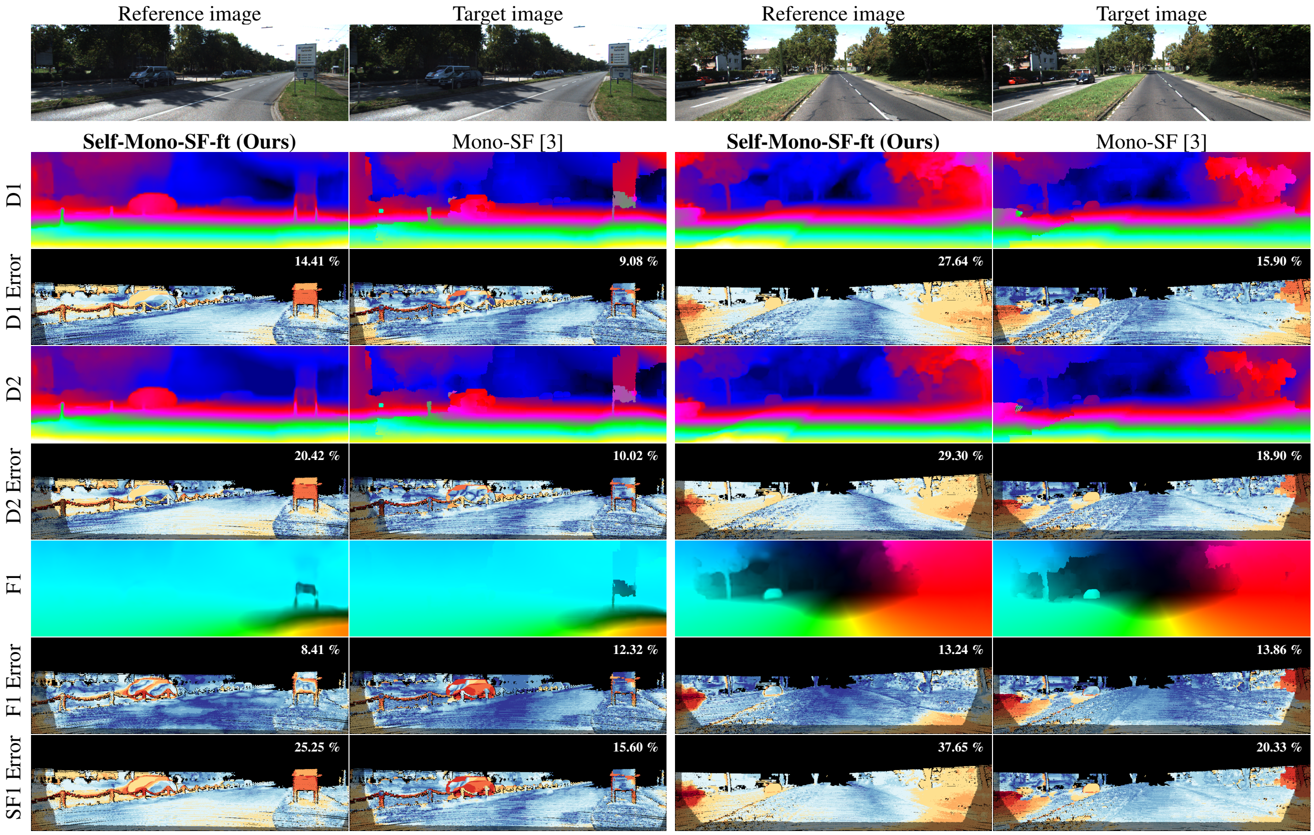


Figure 10. **Failure cases and qualitative comparison with the state of the art on the KITTI 2015 Scene Flow public benchmark [36, 37]**. In the first row, we show two input images, the reference and target image. From the second to the last row, we give a qualitative comparison with Mono-SF [3]: the disparity map of the reference image *(D1)* with its error map *(D1 Error)*, disparity estimation at the target image mapped into the reference frame *(D2)* with its error map *(D2 Error)*, optical flow *(F1)* with its error map *(F1 Error)*, and the scene flow error map *(SF1 Error)*. The outlier rates are overlayed on each error map.